Foundations of Music Warehouses for Discovering New Songs “I like”

Deliege, Francois

Published in:
Proceedings of the 33rd International Conference on Very Large Data Bases

Publication date:
2007

Document Version
Publisher's PDF, also known as Version of record

Foundations of Music Warehouses for Discovering New Songs "I like"*

François Deliège
Supervised by Torben Bach Pedersen
Department of Computer Science
Aalborg University
Denmark
fdeliege@cs.aau.dk

ABSTRACT

Music recommendation systems have become very popular in the recent years. While their first generation was only based on a few manually extracted musical descriptors, the new generation will provide richer and more accurate metadata over very large volumes of music and will be able to generate personalized playlists automatically. These new major improvements call for the development of innovative data management techniques. We propose to create Music Warehouses, dedicated data warehouses optimized for the storage and analysis of music content. We take three major challenges: *(i)* to provide a framework for song similarity searches with respect to different criteria and for the aggregation of low level metadata into high level metadata accessible by end-users, *(ii)* to create advanced dimensions able to support fuzzy hierarchies and versioned hierarchies, and *(iii)* to adequately pre-compute searches to increase the performance of the descriptors analyses. Our current work addresses nearest neighbor searches and the generation of personalized user playlists.

1. INTRODUCTION

The generalization of fast Internet connections and the development of new digital music formats have led to the explosion of digital music available on-line. Large on-line music stores and on-line radios are becoming increasingly popular. This trend is creating a high demand for applications able to organize and search in large music databases. However, current search tools still remain very limited: popular search engines only provide searches based on external annotations, but to offer truly natural and intuitive information retrieval, search and query into the primary media is required. These needs have given a further impulse to the development of new methods for music information retrieval and research on digital music databases.

The Music Information Retrieval (MIR) community has put a lot of efforts into defining and extracting music features, referred to as *low level musical metadata*. Musical metadata is classified into four categories depending on the source from which the information was extracted: acoustic metadata, editorial metadata, cultural metadata and physical metadata [4, 17]. New similarity functions using the musical metadata are constantly being defined; they allow pieces of music to be compared with respect to specific aspects. Unfortunately, the extracted metadata is rarely understandable by regular end-users and in order to be useful, more accessible and meaningful descriptors are needed. At the opposite edge on the scale of music descriptors stands the *high level musical metadata* that allow organizing music into categories, such as genres. Genres benefit from a greater user understanding but suffer from loose definitions. While various taxonomies of high level metadata exist, no real consensus seems to have emerged on generally accepted definitions. Music retailers, music labels, copyright companies, radio stations, end users, etc., have all designed their own taxonomies. Even among the same group of interested parties, e.g., online music portals, inconsistencies are frequent. This situation is source of confusion and finding "new songs that I like" often turns into a long crusade.

The ongoing Ph.D. project focuses on providing dedicated data warehouses optimized for the analysis of music content, referred to as *Music Warehouses* (MWs). MWs will offer the advanced framework necessary to support tools suited to bridge the semantic canyon between low level descriptors and high level contextual representations that carry richer semantics. This goal can be achieved by developing appropriate data structures and enabling analyses of low level descriptors in order to aggregate them into semantically richer representations. The challenges are threefold. First, in order to obtain a better understanding of what types of analyses are appropriate, a survey of existing music management systems was conducted and a case study was built. As a result, we decided to focus on providing automatic playlist generation based on nearest neighbor search analysis. Second, to support such analyses, the problem of modeling advanced data types, which are inherent to the music domain, must be considered. Additionally, new types of dimension hierarchies have to be developed. Fuzzy hierarchies raise the question on how to aggregate data with various degrees of fuzziness. Furthermore, techniques enabling multiple versions of a sin-

*This project is funded by the Danish Research Council for Technology and Production Sciences, through the framework project "Intelligent Sound", grant no. 26-04-0092.

VLDB '07, September 23-28, 2007, Vienna, Austria.

gle dimension hierarchy have to be investigated. Third, the issue of data pre-computation must be addressed in order to increase the performance of the analysis while preserving the wide heterogeneity of high level music representations. This issue is of critical importance given the high computational requirements of extracting and calculating similarity values between musical descriptors.

The ultimate goal of the project is to build the first publicly available MW and to integrate it into the platform developed by the Intelligent Sound project[1]. The dissertation has so far one year of development and a research plan has been drawn for the next two years. A first prototype of MW is currently being built and consists of 60000 songs and their corresponding extracted features. The database was constructed with the data gathered using the MIRocket[2] Winamp plugin and proper tag information was verified against the Music Brainz database[3].

The remainder of this paper is organized as follows. We present related music search databases approaches and music information retrieval work in Section 2. Section 3 provides more details on previous and ongoing work on the MW. We proceed by discussing the research challenges building an MW offers in Section 4. Finally, we conclude in Section 5.

2. RELATED WORK

Today, most of the music recommendation systems available online are solely based on editorial information available through tags or on collaborative filtering. With the growth of the Web, techniques based on publicly available data have emerged. They combine data from many sources using text analysis and collaborative filtering techniques to determine similarity. Since they are based on human opinion, these approaches capture many cultural and other intangible factors. A main disadvantage of these techniques, however, is that they are only applicable to music for which a reasonable amount of reliable public information is available. MoodLogic[4] is an example of how editorial information and cultural information can be gathered. The core idea of MoodLogic is to associate metadata to songs automatically, thanks to two basic techniques: first, an audio fingerprinting technology able to recognize music titles on personal hard disks, second, a database collecting user ratings on songs, which is incremented automatically in a collaborative fashion. No acoustic analysis of the song is, however, performed. Pandora[5] adopts a radically different approach and bases its recommendations on acoustic metadata. The acoustic metadata is generated through the music genome project that consists of a small group of music experts manually describing the audio content of songs. While these systems provide useful services for Electronic Music Distribution (EMD) systems, their scalability remains limited as they rely only on expensive and external sources of information.

Many researchers have studied the music similarity problem by analyzing symbolic representations such as MIDI music data, musical scores, and the like. For example, methods have been developed to search for pieces of music with a particular melody. The queries can be formulated by humming and are usually transformed into a symbolic representation, which is matched against a database of scores, e.g., in MIDI format [10]. However, techniques based on MIDI or scores are limited as they are format dependent. Acoustic representation allows music content to be directly analyzed and can, therefore, be applied to any music. A considerable amount of work has been reported on automatic extraction of audio features [12]. For monophonic music, Downie indexed with good results a database of 10000 songs by adapting existing text information retrieval techniques to music [8]. Blum et al. [2] present an indexing system based on matching features such as pitch, loudness or Mel-frequency cepstral coefficients, briefly MFCC. Tzanetakis extracts a variety of features representing the spectrum, rhythm and chord changes and concatenates them into a single vector to determine similarity [24].

Research on distributed music database in P2P and Wireless Ad-hoc networks was conducted by Karydis et al. [13]. They also have developed an algorithm that efficiently retrieves audio data similar to an audio query. The proposed method utilizes a feature extraction technique for acoustical music sequences [14]. Downie et al.[7] describe a secure and collaborative framework for evaluating music information retrieval algorithms but little attention has been paid so far to the storage issues of audio features in data warehouses. A more traditional approach is to use classical relational models such as the one proposed by Rubenstein that extends the entity-relationship data model to implement the notion of hierarchical ordering, commonly found in musical data [20]. Through some examples, Rubenstein illustrates how to represent musical notation in a database using the extensions he introduces, but no detailed data types and operations are given. A multimedia data model, following the layered model paradigm that consists of a data definition layer, a data manipulation layer, a data presentation layer, and a control layer, is presented by Wynblatt et al. [28], but no query language is proposed. Surprisingly, none of these models adopts a multidimensional approach by representing data in cubes, a very convenient structure for performing on-the-fly analysis of large volumes of data that has already proved its strengths in data warehouses [18]. Finally, the most relevant related work is the music data model and its algebra and query language presented by Wang et al. [25]. The data model is able to structure both the musical content and the metadata but does not address performance optimization issues. In particular, it does not provide an adequate framework to perform similarity based search, dimensions do not support hierarchies, and no discussion is provided about the indexing issues.

Existing indexing techniques, such as M-grid [6] or M-tree [3], can be applied to index high dimensional musical feature representations. However, as a consequence of the subjective nature of musical perception, the triangular inequality property of the metric space is typically not preserved for similarity measures [1, 16, 19]. Therefore, additional techniques have to be employed to ensure a suitable foundation for musical similarity search. Nearest neighbor searches are a popular topic in the database community for their usage in content based retrieval and similarity searches. An impressive amount of work can be found in the literature for both high and low dimensional spaces [15, 22, 26, 30]. However, these techniques are inspired by geometric problems and rely on the existence of a Euclidian space

[1]http://www.intelligentsound.org
[2]http://www.intelligentsound.org/demos/mirocket.htm
[3]http://musicbrainz.org
[4]http://www.moodlogic.com
[5]http://www.pandora.com

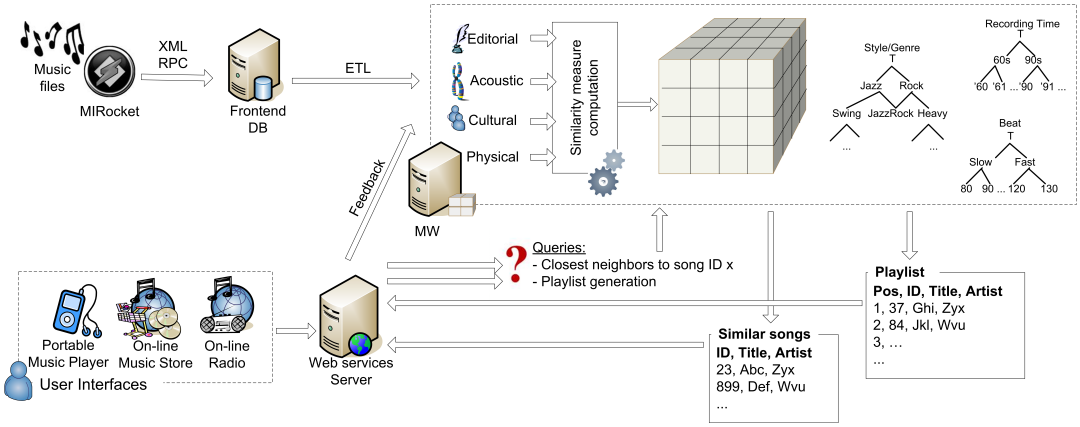

Figure 1: Overview of the Music Recommendation System Architecture

in order to eliminate potential candidates using upper and lower bounds. Unfortunately, they are not suitable for non-metric space and are therefore not applicable for the similarity measures where the triangular inequality property is not fulfilled. Work on indexes for non-metric space is presented in the literature [11, 21, 23, 29]. Though the similarity function is non-metric, it remains confined in a pair of specifically constructed lower and upper bounds. However, as no consensus has yet been reached in the MIR community, our aim is to create MWs as parametrized frameworks, able to work simultaneously with several and user-defined similarity functions.

3. OUR APPROACH

In this section, we provide more details on our previous and ongoing work on the MW. We first propose a short usage scenario, then present the overall architecture of the music recommendation framework and discuss our solutions for answering song similarity queries and manipulating playlists.

3.1 Scenario

We propose to develop a on-line music recommendation system able to find the song the most similar to another with respect to one or a combination of musical aspects. The musical aspects should represent high level metadata accessible to users such as the beat or the genre of a song. They should not only be based on audio analysis but also on cultural elements such as the targeted audience, on editorial aspects such as the artists, or on physical information such as the type of format. Furthermore, the MW has to handle user playlists and offer playlist manipulation tools. Eventually, we want to offer to users an automatic music playlist generation service. The playlist should be created with respect to constrains such as the user's profile. Additionally, the user could specify the length of the playlist and how the playlist should evolve.

3.2 Architecture Overview

Figure 1 shows our prototypical system architecture. The system is composed of a client and a server. The client is responsible for extracting new music features, for building the user profile, and for all the user interface related issues. The server is responsible for centralizing music information, for storing the user profile, for finding similar songs, and for generating playlists. First, the music not yet present on the server gets analyzed and the low level features are extracted and transmitted to the server. Based on the low level features, computationally expensive similarity value are generated. Second, once the data cubes are generated, queries to obtain the songs the most similar to a given seed song or to generate a playlist can be answered. User feedbacks on the provided results are saved for a future integration in the MW.

3.3 Song Similarities

The music recommendation is able to compare and provide a similarity value for any pair of songs. Using the similarity values, it is easy to tell if the two songs of any given pair of songs are "very different", "different", "somewhat similar", or "very similar" from the perspective of a given attribute such as the beat for example. But, in order to keep our scenario as general as possible, as few assumptions as possible should be made about the properties of the similarity functions. In particular, the similarity aspects do not have to be a metric in a mathematical sense (positive, triangular inequality, and symmetry properties). However, we assume that the similarity values can be mapped to values in the interval from 0 to 1.

For each song, referred to as the seed song, the similarity values with the other songs are computed and stored in a fuzzy set [5]. The similarity value between two songs is then represented using a membership degree, e.g., the more similar two songs are, the closer to 1 the membership degree will be. To retrieve the closest songs to a given song seed, the MW reads the fuzzy set and returns the songs having the highest membership degree. While this solution offers fast retrieval of the most similar songs regardless of the properties of the similarity functions, it also suffers from major drawbacks namely the storage consumption and the updating process.

We are currently extending the fuzzy set algebra presented by Galindo et al [9]. We have created operators to perform Top_k selections, reductions, and average aggregations among fuzzy sets. The Top_k operator changes the membership degree to zero to all elements that are not part of the k elements having the highest degree. The reduction operator changes to 0 all membership degrees below a given threshold. As an example, Figure 2 shows the aggregation

of two song sets using the intersection operator. Using the fuzzy set algebra, it is possible to execute complex queries such as retrieving the songs that are the most similar to two given song seeds, or to one song seed with respect to different similarity aspects.

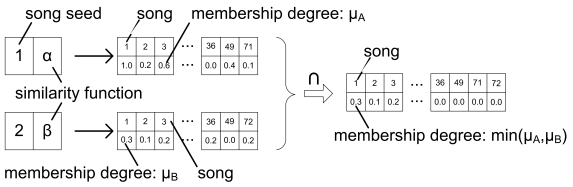

Figure 2: Intersection of Fuzzy Sets of Closest Songs

Additionally, we are now focusing on query optimization aspects of these operators for the different physical storage options. We have studied the use of tables, arrays and WAH compressed bitmaps [27] as internal storage solutions for the fuzzy sets and discussed the advantages of each. Tables are space consuming but allow pivoting between songs seeds and songs in the fuzzy set. The choice between arrays and compressed bitmaps depends on the number of bits required to identify uniquely an element of the fuzzy set and to store the membership degrees. The choice is, therefore, depending on the data set.

We envision the use of fuzzy sets in two additional cases. First, fuzzy sets can be used to represent users' opinion about the previously suggested songs. It is possible to retrieve the set of the proposed songs which a user liked or disliked in a particular session. Second, regardless of the context, e.g., the previously played songs, the suggested songs, or the criteria used, a user is able to grade if he likes a song or not. For example, a song banned by the user should never be played. To the contrary, other songs should be proposed more often as the user likes them. The list of songs a given user likes, and the songs he dislikes can also be stored using a fuzzy set. The MW allows aggregation to be performed using the favorite songs fuzzy set attribute of the user dimension, e.g., counting the users having marked a particular set of songs as their favorites.

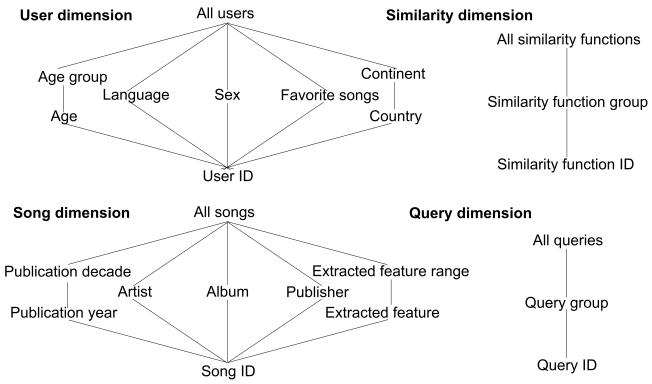

Figure 3: Dimension Hierarchies

The closest songs cube provides a set of songs which are the closest to a given one, referred to as the seed song, with respect to a similarity function, so that, for each song, for each similarity function, the closest songs are stored using a fuzzy set. It is composed of the song and the similarity dimensions as shown in Figure 3. The user feedback cube gathers relevance statistics about the songs proposed to users by the music recommendation system. It is composed of the user dimension and the query dimension. For each user and query, the user feedback is stored. The feedback given for a particular song played is stored as a membership degree representing how the proposed song is pertinent in the context of the query. A very low membership degree is given when the user believes the song should not have been proposed.

3.4 Generic Playlists

We want to build, through user collaborative filtering, playlists composed of a given number of songs and a theme. For example, 100 users have to build a playlist that is made of ten songs, that is composed of 3 rocky songs, 3 jazzy songs, 3 romantic songs and 1 blues song. Furthermore, the users are asked to respect some smooth transitions between the songs so that the third rocky song sounds a bit jazzy. Finally, let's also assume that independently from the playlist building process each registered user has a list of songs he likes and dislikes.

The playlists created by the voting users are merged into a unique playlist, referred to as a *generic playlist*, that can be shared among all the users registered in the system. However, some users might be disappointed by the generic playlist if inserted blindly to their music player. For example, the user might have previously banned some songs and would prefer the system to find alternatives. To the contrary, if a song was very close from being selected as part of the playlist and the user rated it as his favorite song, the system should switch the song originally present in the generic playlist with the user's favorite song. Figure 4 illustrates the construction of a generic playlist and how it can at be derived into a personalized playlist a later stage.

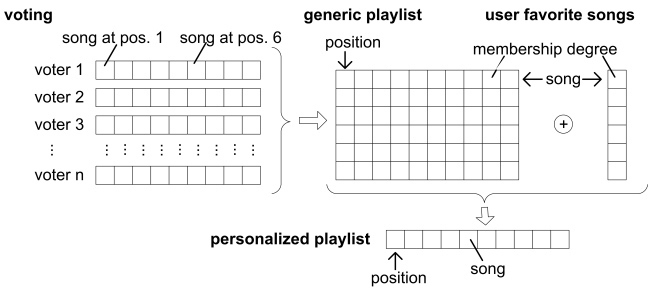

Figure 4: Generic Playlists Usage Scenario

Generic playlists offer a concrete scenario for the usage of *fuzzy lists*, the fuzzy counterpart of the well-known (crisp) lists. We propose to define a finite fuzzy list, A, of size m, over a domain of discourse, denoted X, as follows:

$$A = \{\mu_A(x,n)/n/x : \\ x \in X, n \in \{1,\ldots,m\}, \mu_A : X \times \mathbb{N} \mapsto [0,1]\}$$

where x is an element of X, where n is a non-negative integer, and where $\mu_A(x,n)$, referred to as the sequential membership degree of x at position n, is a real number belonging to $[0,1]$, with $\mu_A(x,n) = 0$ when x does not belong to A at position n, and $\mu_A(x,n) = 1$ when x completely belongs to

A at position n. The fuzzy list concept is new and new operators such as the concatenation, the intersection, the union, the subset, the inverse, and the negation have to be formally defined and studied. We believe fuzzy lists are a superset of both fuzzy sets and classical crisp lists theories, they could therefore be used in a large scope of applications. We are currently conducting further investigations on the storage and the query processing issues fuzzy lists introduce.

4. NEXT CHALLENGES

During the first year of the Ph.D. project, a case study has been initiated by analyzing complex real-world music data taken from various music data extraction and management systems. In parallel, concepts and techniques underlying the system have been studied and have led to the identification of new research challenges for the database community [4]. We present below the three research topics that will be pursued in the remainder the PhD project. They add significant functionalities to Section 3 and will be addressed in the following order as time permits. For each, query processing and optimization techniques such as the usage of materialized views and pre-aggregation will be studied.

4.1 Fuzzy Hierarchies

Non-strict hierarchies, i.e., hierarchies supporting elements having multiple parents, allow different paths to be followed when performing roll-up operations. In the time dimension, days can be rolled-up into months and in turn into years. Similarly, days can be rolled-up into weeks by following a different path since there are overlaps between week–month and week–year precision levels. While non-strict hierarchies are useful, they seem sometimes very artificial. In the time dimension example, a linear aggregation path composed of overlaps, dayweekmonthsyears, appears more intuitive.

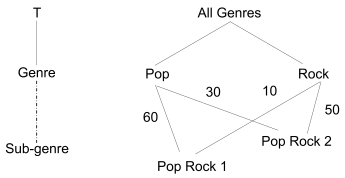

Figure 5: Fuzzy Hierarchy of the Genre Dimension

The expected contribution of the PhD project will consist in the creation of a data model and its algebra enabling the use of *fuzzy hierarchies*. Fuzzy hierarchies enable "children" to belong to multiple parents with various degrees of affiliation evaluated through membership functions as defined by fuzzy logic. While fuzzy hierarchies are not required to handle typical data warehousing demands, they become unavoidable for MWs in order to represent complex hierarchies such as in the genre dimension as illustrated in Figure 5. Sub-genres would belong to genres to a certain degree, e.g., the genre Jazz-Rock could belong 60% to Jazz and 40% to Rock, a notion that multidimensional data models have not been able to fully capture so far. Moreover, issues on modeling fuzzy hierarchies will be compared with current work on ontologies.

4.2 Versioned Hierarchies

In a multidimensional database, a dimension hierarchy is said to be: *strict*, if all dimension values have no more than one direct parent, *onto*, if the hierarchy is balanced, and *covering*, if no containment path skips a level.In a MW, the hierarchy of the genre dimension is non-strict, non-onto and non-covering. However, this is not sufficient.

Since very little consensus exists on taxonomies, the techniques already in place for slowly changing dimensions in multidimensional databases may not be appropriate. Instead, MWs require support for versioning abilities, mimicking software versioning systems such as CVS or Subversion, where different hierarchies can coexist and evolve.

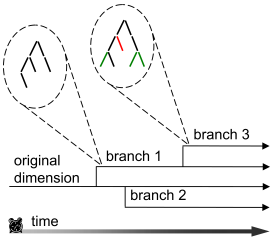

Figure 6: Versioned Hierarchies

We expect the Ph.D. project to contribute in the creation of a data model and algebra enabling the use of *versioned hierarchies*. Figure 6 shows a versioned genre hierarchy that, for example, defines a classification of the genre dimension for different user profiles. Versioned hierarchies call for the creation of new database operators. Examples of such operators are: navigation in the different branches of the hierarchy, comparison of the branches between users, evolution of a user hierarchy over time, and ability to derive branches from existing ones.

4.3 Precision Aware Retrieval

Certain queries performed in an MW do not require exact answers. Rather, rough approximations would be sufficient. For example, nearest neighbor queries, such as the ranking of the k nearest neighbors of a given song, do not focus on the exact position of each song compared to a given one, but rather on coarser notion of distance, such as very close, close, or far. The exact granularity of the answer should not be fixed but rather determined either implicitly by an appropriate algebra, or explicitly in the query. Queries including the notion of ranking, referred to as Top-K queries, are very frequent in data warehouses. At the query processing level, optimizations can be performed in order to drastically improve the response time. Operators such as ranked selection and ranked joins use specific algorithms that have already demonstrated their usefulness for relational models.

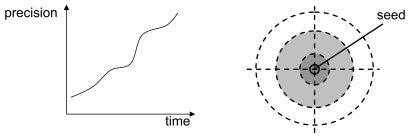

Figure 7: Precision Aware Retrieval

In MWs, however, Top-K queries require ranking at a coarse level of granularity where elements need to be ordered only in subsets, e.g., very close, close and far. Another aspect of quality-of-service in MWs is the response time. Time consuming queries, such as music comparison and nearest

neighbor searches, spark the need for new techniques able to trade fast response time for precision. Query answers need to take the form of streams, updated progressively with more precise and reliable information as pictured in Figure 7. Our contribution will consist in developing an algebra and query processing techniques to enable both coarse Top-K queries and *precision improvement streams*. For example, asking what the common characteristics of a set of songs are could result in the immediate creation of a stream of music characteristics in their high-level representation, starting with coarse similarities and progressively refining similarities as the query processing continues.

5. CONCLUSION

Music recommendation systems intensively rely on metadata extraction and similarity functions. However, to obtain valuable recommendations, extraction and similarity functions become computationally expensive. In parallel, new features such as the generation of personalized playlists, playlists sharing, and collaborative playlist creation are appearing. This calls for the development of efficient data management solutions for the computed similarity values and customized playlists. In this paper, we have presented an MW prototype to tackle these issues. First, we proposed to store similarity values using fuzzy sets. Second, we introduced fuzzy lists to manipulate generic playlists. Our current research is focused on creating an adequate algebra, and addressing storage, indexing and query processing issues for both fuzzy sets and fuzzy lists. Finally, we have presented three research challenges that will be investigated at a later stage in the project. We expect the project to generate valuable results profitable to both the music information retrieval and the database communities.

6. REFERENCES

[1] J.-J. Aucouturier, F. Pachet, and M. Sandler. The way it sounds : Timbre models for analysis and retrieval of polyphonic music signals. *IEEE Trans. of Multimedia*, 7(6), 2005.

[2] T. L. Blum, D. F. Keislar, J. A. Wheaton, and E. H. Wold. Method and article of manufacture for content-based analysis, storage, retrieval, and segmentation of audio information. *U.S. Patent 5, 918, 223*, 1999.

[3] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proc. of VLDB*, 1997.

[4] F. Deliège and T. B. Pedersen. Music warehouses: Challenges for the next generation of music search engines. In *Proc. of LSAS*, 2006.

[5] F. Deliège and T. B. Pedersen. Fuzzy song sets for music warehouses. In *Proc. of ISMIR*, 2007.

[6] C. Digout and M. A. Nascimento. High-dimensional similarity searches using a metric pseudo-grid. In *Proc. of ICDEW*, 2005.

[7] J. S. Downie, J. Futrelle, and D. K. Tcheng. The international music information retrieval systems evaluation laboratory: Governance, access and security. In *Proc. of ISMIR*, 2004.

[8] J. S. Downie and M. Nelson. Evaluation of a simple and effective music information retrieval method. In *Proc. of ACM SIGIR*, 2000.

[9] J. Galindo, M. Piattini, and A. Urrutia. *Fuzzy Databases: Modeling, Design and Implementation*. Idea Group Pub, 2005.

[10] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query by humming: Musical information retrieval in an audio database. In *Proc. of ACM Multimedia*, 1995.

[11] K.-S. Goh, B. Li, and E. Chang. DynDex: a dynamic and non-metric space indexer. In *Proc. of ACM Multimedia*, 2002.

[12] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 2003.

[13] I. Karydis, A. Nanopoulos, A. Papadopoulos, D. Katsaros, and Y. Manolopoulos. Content-based music information retrieval in wireless ad-hoc networks. In *Proc. of ISMIR*, 2005.

[14] I. Karydis, A. Nanopoulos, A. Papadopoulos, and Y. Manolopoulos. Audio indexing for efficient music information retrieval. In *Proc. of MMM*, 2005.

[15] F. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, and Z. Protopapas. Fast nearest neighbor search in medical image databases. In *Proc. of VLDB*, 1996.

[16] B. Logan and A. Salomon. A music similarity function based on signal analysis. In *Proc. of ICME*, 2001.

[17] F. Pachet. *Knowledge Management and Musical Metadata*. Idea Group, 2005.

[18] T. B. Pedersen and C. S. Jensen. Multidimensional database technology. *IEEE Computer*, 34(12), 2001.

[19] T. Pohle, E. Pampalk, and W. G. Generating similarity-based playlists using traveling salesman algorithms. In *Proc. of DAFx*, 2005.

[20] W. B. Rubenstein. A database design for musical information. In *Proc. of ACM SIGMOD*, 1987.

[21] M. M. Ruxanda and C. S. Jensen. Efficient similarity retrieval in music databases. In *Proc. of COMAD*, 2006.

[22] T. Seidl and H. Kriegel. Optimal multi-step K-nearest neighbor search. In *Proc. of ACM SIGMOD*, 1998.

[23] S. Sheu and J.-R. Wu. GC*-tree: a generic index for perceptual similarity search. In *Proc. of ITRE*, 2005.

[24] G. Tzanetakis, G. Essl, and P. R. Cook. Automatic musical genre classification of audio signals. In *Proc of ISMIR*, 2001.

[25] C. Wang, J. Li, and S. Shi. A music data model and its application. In *Proc. of MMM*, 2004.

[26] R. Weber, H. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensinal spaces. In *Proc. of VLDB*, 1998.

[27] K. Wu, E. J. Otoo, and A. Shoshani. Optimizing bitmap indices with efficient compression. *ACM Trans. of Database Systems*, 31(1), 2006.

[28] M. J. Wynblatt and G. A. Schloss. Control layer primitives for the layered multimedia data model. In *Proc. of ACM Multimedia*, 1995.

[29] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proc. of ICDE*, 1998.

[30] C. Yu, B. Ooi, K. Tan, and H. Jagadish. Indexing the distance: An efficient method to KNN processing. In *Proc. of VLDB*, 2001.